|  |  |
| --- | --- |
| 抽象基类 | 节点流（文件流） |
| InputStream | FileInputStream |
| OutputStream | FileOutputStream |
| Reader | FileReader |
| Writer | FileWriter |

**IO体系**

**字节流**

* **FileInputStream**
* 从硬盘中存在的一个文件中，读取其内容到程序中。要求：读取的文件一定要存在。
* 使用try-catch的方式处理异常更合理：这样保证流的关闭操作一定可以执行。
* read()方法：读取文件的一个字节。当执行到文件结尾时，返回-1。
* 步骤：
* 创建一个File类的对象；
* 创建一个FileInputStream类的对象；
* 调用FileInputStream的read()方法，实现file文件的读取；
* 关闭输入流。

@Test  
 public void test3() {  
 File file = new File("resources/hello.txt");  
 FileInputStream fis = null;  
 try {  
 fis = new FileInputStream(file);  
  
 // 读取到的数据要写入数组  
 byte[] b = new byte[5];  
 // 每次读入到byte中的字节的长度  
 int len;  
 while ((len = fis.read(b)) != -1) {  
// for (int i = 0; i < len; i++) {  
// System.out.print((char) b[i]);  
// }  
  
 String str = new String(b, 0, len);  
 System.***out***.print(str);  
 }  
 } catch (IOException e) {  
 e.printStackTrace();  
 } finally {  
 //关闭输入流

if (fis != null) {  
 try {  
 fis.close();  
 } catch (IOException e) {  
 e.printStackTrace();  
 }  
 }  
 }  
 }

* **FileOutputStream**
* 步骤：
* 创建一个File类的对象，表明要写入的文件位置；
* 创建一个FileOutputStream类的对象，将file的对象作为形参传递给FileOutputStream的构造器中；
* 写入操作；
* 关闭输出流。
* 注：输出的物理文件可以不存在，当执行过程中，若不存在，会自动创建。若存在，会将原有的文件覆盖。

@Test  
public void test4() {  
   
 File file = new File("hello2.txt");  
 FileOutputStream fos = null;  
  
 try {  
 fos = new FileOutputStream(file);  
 //写入操作  
 fos.write(new String("I love China!").getBytes());  
 } catch (Exception e) {  
 e.printStackTrace();  
 } finally {  
 //关闭输出流  
 if (fos != null) {  
 try {  
 fos.close();  
 } catch (Exception e) {  
 e.printStackTrace();  
 }  
 }  
 }  
}

* **从硬盘读取一个文件，并写入到另一个位置。（相当于文件的复制）**
* 步骤：
  + 提供读入、写出的文件；
  + 提供相应的输入、输出流；
  + 实现文件的复制；
  + 关闭输出流、输入流。

@Test  
public void test5() {  
   
 File file1 = new File("hello.txt");  
 File file2 = new File("hello3.txt");  
 FileInputStream fis = null;  
 FileOutputStream fos = null;  
  
 try {  
 fis = new FileInputStream(file1);  
 fos = new FileOutputStream(file2);  
  
 //实现文件的复制  
 byte[] b = new byte[20];  
 int len;  
 while ((len = fis.read(b)) != -1) {  
 fos.write(b, 0, len);  
 }  
 } catch (Exception e) {  
 e.printStackTrace();  
 } finally {  
 //关闭输出流  
 if (fos != null) {  
 try {  
 fos.close();  
 } catch (IOException e) {  
 e.printStackTrace();  
 }  
 }  
 //关闭输入流  
 if (fis != null) {  
 try {  
 fis.close();  
 } catch (IOException e) {  
 e.printStackTrace();  
 }  
 }  
 }  
}